확장된 MapReduce 를 이용한 병렬 진화 전략

최현화*, 이미영*, 이규철**
*한국전자통신연구원 데이터베이스연구팀
**충남대학교 데이터베이스연구실
e-mail : hyunwha@etri.re.kr

Parallel Evolution Strategy Using an Extended MapReduce

Hyun Hwa Choi*, Mi Young Lee*, Kyu Chul Lee*
*Database Research Team, Electronics and Telecommunications Research Institute
**Dept. of Computer Engineering, Chung-Nam University

요 약

진화 전략은 생식, 돌연변이, 재조합과 같은 생물의 진화과정을 모델링하여 복잡한 문제를 해결하고자 하는 개체군 기반의 조합 최적화 알고리즘 중의 하나이다. 데이터 집약적이며, 소요 시간이 오래 걸리는 진화 전략은 클라우드 컴퓨팅 하의 IT 서비스로서 적합한 대표적인 예이다. 이에 본 논문에서는 최근 분산 환경 하에서 병렬 처리 응용을 쉽게 개발할 수 있도록 지원하는 프로그래밍 모델인 MapReduce 를 확장하여 진화 전략을 수행할 수 있는 방법을 제안한다.

1. 서론

진화 알고리즘(evolution algorithms; EAs)은 생식, 돌연변이, 재조합과 같은 생물의 진화과정을 모델링하여 복잡한 문제를 해결하고자 하는 개체군 기반의 조합 최적화 알고리즘(population-based combinatorial optimization algorithm) 이다. 진화 알고리즘은 유전자 알고리즘, 진화 전략, 진화 프로그래밍, 유전 프로그래밍 등을 포함한다. 특히, 진화 전략은 실수 벡터를 사용하는 국소해가 많은 최적화 문제의 해를 구하는 것을 목적으로 개발되었으며, 데이터 집약적이며, 소요 시간이 많이 걸리는 작업이라 하겠다. 그리하여 지금껏 꾸준한 병렬화 연구가 있어 왔으나, 각 컴포넌트 간의 통신 및 동기화 등 분산 환경 하에서 개발상의 어려움이 여전이 존재하는 것이 현실이다.

최근 데이터 센터 내에서 분산 응용의 용이한 개발을 지원하는 프로그래밍 모델인 MapReduce[1]가 구글(Google)에 의해 처음 소개되었다. 야후(Yahoo!) 및 마이크로소프트는 MapReduce 와 유사한 시스템을 개발 중에 있고[2][3], 다양한 환경에서 MapReduce 를 활용하기 위한 연구도 많이 이뤄지고 있다[4][5][6]. 이에, 본 논문에서는 데이터 센터와 같은 분산 환경 하에서 진화 전략을 병렬 수행하기 위하여 MapReduce 의 모델을 확장하는 방법에 대해서 기술하도록 한다.

본 논문의 구성은 다음과 같다. 2 장에서는 진화 전략에 대해서 간략히 살펴보고, 3 장에서는 MapReduce 의 모델을 살펴보며, 4 장에서는 병렬 진화 전략을 위하여 확장된 MapReduce 를 소개한다. 마지막으로 5 장에서는 결론을 맺는다.

2. 진화 전략

진화 전략은 재조합, 돌연변이, 적합도 평가 그리고 선택의 과정을 반복 수행한다. 초기의 진화 전략은 하나의 개체로부터 집단의 탐색을 행하여, 돌연변이에 의해 만들어진 자손이 부모보다 적합도가 좋게 될 때 다음 세대의 부모로 채택되었으나, 그 후 복수 개체의 집단에 의한 탐색 수법으로서 여러 가지 형태로 발전하여 왔다. 일반적인 진화 전략은 다음 세대를 구성하기 위한 부모 개체 선택법과 교배 사용 유무에 따라 크게 (μ,λ) 진화 전략과 $(\mu+\lambda)$ 진화 전략으로 구분된다.

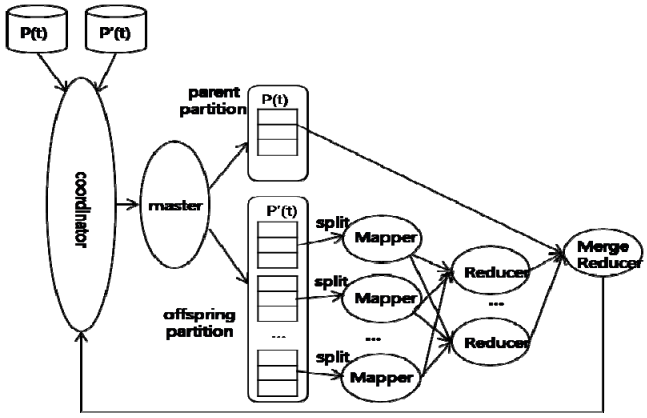
3. MapReduce

구글의 MapReduce 는 분산 환경 하에서 대용량의 데이터 처리를 위하여 제안된 함수 기반 병렬 프로그래밍 기술이다. 다시 말하면, 사용자가 메모리 관리, 병렬화, 네트웍 프로그래밍에 대한 고려없이 키-값의 쌍을 기반으로 한 Map 과 Reduce 함수를 기술하기만 하면, 자동으로 분산 자원의 병렬화 및 스케쥴링을 통해 실행을 보장한다. 구글 파일 시스템(Google File System: GFS)이라는 분산 파일 시스템을 바탕으로, 실행은 두 단계로 이뤄진다. 첫번째 Map 단계에서는 입력 데이터를 받아 계산 후 중간 키-값 쌍을 생산하여, 노드의 로컬 디스크에 저장한다. 저장된 중간 키-값 쌍들은 중간 키를 기반으로 그룹핑되어 Reduce 함수에 전달된다. 두번째 Reduce 단계에서는 입력 받은 중간 키-값을 대상으로 값을 조합하여 최종 결과를 생산한다.

4. 병렬 진화 전략을 위한 MapReduce

유전 알고리즘의 병렬 처리를 위한 연구[4]을 참

고하여, 본 논문에서 제안하는 병렬 진화 전략을 위한 확장된 MapReduce 의 시스템 구조는 그림 1 과 같다.



(그림 1) 병렬 진화 전략을 위한 시스템 구조

Coordinator 는 부모 개체들의 집합으로부터 재조합 및 돌연변이를 통한 자손 개체를 발생 시킨다. Map-Reduce-MergeReduce 를 통해 부모와 자손 개체의 적합도 검사 및 이에 따른 다음 세대가 결정되면 종료 조건 여부를 검사하고, 만족할 때까지 반복 작업을 수행한다. Coordinator 의 상세한 수행 알고리즘은 다음과 같다.

procedure MapReduceMerge_Parallel_ES()
t=0
P[0] = a1[0], ..., au[0]
evaluate(a1[0], ..., au[0])
while not T(P[t]) do
P'[t] = mutate(recombine(P[t])
sendToMaster(P[t], P'[t], es-type)
P[t+1] = ReceiveFromMaster(t)
t= t+1
endwhile
return P[t]

Mapper 는 대용량의 개체에 대한 적합도 검사를 수행한다. 즉, 다음 세대 결정을 위하여 이미 적합도 검사가 끝난 부모 개체는 대상이 되지 않는다.

function Mapper (key, value)
$P=a_1[0], ..., a_u[0] = individual(value)$
$P = evaluate(a_1[0],.., a_n[0])$
emit(default_key, P)

Reducer 는 Mapper 로부터 발생한 자손 개체들로부터 지역적인 최적의 해를 계산해 낸다. Reducer 는 단지 새로이 생성된 자손 개체에 한하여 수행한다. 이는 진화 전략에서는 부모 개체는 자손 개체에 비해 그 수가 적고, 항상 같은 수를 가진다고 가정하고 있기 때문이다.

function reducer(key, value_list)
i=0
foreach value in value_list
$P[i]=a_1[i], ..., a_u[i] = individual(value)$
i++
P' = select(P)
foreach individual in P'
emit(individual, 1)

MergeReducer 는 분산 파일 시스템에 존재하는 부모 개체의 적합도 값과 Reducer 로부터 얻어지는 노드 내 지역 디스크에 저장된 자손 세대 최적의 해(적합도 검사 값)을 대상으로 가장 최적의 해를 구한다.

Function mergeReducer(key, value)
$P=a_1[i], ..., a_u[i] = individual(key)$
Emit(P, 1)

5. 결론

본 논문에서는 데이터 센터 내에서 대용량의 데이터를 대상으로 진화 전략을 병렬적으로 수행하기 위하여, 기존의 MapReduce 모델을 확장한 Map-Reduce-MergeReduce 의 시스템 구조를 제안하였다. 특히 MergeReduce 는 분산 파일 시스템 내에서 부모 개체의 적합도 값과 Reduce 로부터 발생한 노드의 로컬 디스크에 저장된 자손 세대의 최적의 해를 대상으로 가장 최적의 해를 구하도록 설계되었다. 본 논문에서 제안한 확장 MapReduce 를 이용한 진화 전략의 병렬 처리에 대한 성능 수치는 다음 논문에서 제시하도록 한다.

참고문헌

[1] Jeffrey Dean and Sanjay Ghemawat. “MapReduce: Simplified Data Processing on Large Clusters”, proceeding of the USENIX Symposium on operation Systems Design and Implementation, 2004
[2] http://hadoop.apache.org/mapreduce/
[3] Michael Isard, Mihai budiu, Yuan Yu, Andrew Birrell and Ennis Fetterly. “Drayd: Distributed Data Parallel Programs from Sequential Building Blocks”, proceeding of the EruoSys Conference, 2007
[4] Chao Jin, Christian Vecchiola and Rajkumar Buyya. “MRPGA: An Extension of MapReduce for Parallelizing Genetic Algorithm”, proceeding of the 4th IEEE International Conference on e-Science, 2008
[5] Hung-chih Yang , Ali Dasdan, Ruey-Lung Hsiao and D. Stott Parker. “Map-Reduce-Merge: Simplified Relational Data Processing on Large Clusters”, proceeding of the SIGMOD, 2007
[6] Jaliya Ekanayake and Shrideep Pallickara. “MapReduce for Data Intensive Scientific Analyses”, proceeding of the 4th IEEE International Conference on e-Science, 2008

본 연구는 정보통신부 및 정보통신연구진흥원의 IT 신성장동력핵심기술개발사업의 일환으로 수행하였음. [2007-S-016-1, 저비용 대규모 글로벌 인터넷 서비스 솔루션]